Pruning for GNNs: Lower Complexity with Comparable Expressiveness

Dun Ma[1] Jianguo Chen[2,4] Wenguo Yang[2] Suixiang Gao[2,5]
Shengminjie Chen[3]

[1]School of Advanced Interdisciplinary Sciences, University of Chinese Academy of Sciences
[2]School of Mathematical Sciences, University of Chinese Academy of Sciences
[3]Institute of Computing Technology, Chinese Academy of Sciences
[4]Academy of Mathematics and Systems Science, Chinese Academy of Sciences
[5]Zhongguancun Laboratory.Beijing, China

May 25, 2025

Outline

Graph Neural Network

- Standard Message-Passing GNN:

$$M_v^l = AGG^l(\{\!\{H_u^{l-1} | u \in N(v)\}\!\}) \tag{1}$$

$$H_v^l = COB^l(H_v^{l-1}, M_v^l) \tag{2}$$

- Multi-Aggregation GNN:

$$M_v^{l,k} = AGG_k^l\left(\{\!\{H_u^{l-1} \mid u \in N_{path}^k(v)\}\!\}\right), \tag{3}$$

$$\mathbf{M}_v^l = (M_v^{l,1}, M_v^{l,2}, \cdots, M_v^{l,K}) \tag{4}$$

$$H_v^l = COB^l\left(H_v^{l-1}, \mathbf{M}_v^l\right) \tag{5}$$

Matrix Language

Theorem

Two adjacency matrices are indistinguishable by the 1-WL test if and only if $e(A_G) = e(A_{G'})$ for all $e \in ML(\mathcal{L}_1)$, where $\mathcal{L}_1 = \{.,^\top, \mathbf{1}, \mathrm{diag}\}$.

- Matrix languages $ML(\mathcal{L})$ can be formalised through composition of linear algebra operations. A linear algebra operation takes a number of matrices as input and returns another matrix (vector or scalar).
- For example, if A is a adjacency matrix, then $e(A) = \mathbf{1}^\top A\mathbf{1}$ is a scalar sentence in $ML(\mathcal{L})$ with $\mathcal{L} = \{.,^\top, \mathbf{1}\}$, computing the number of edges in G.
- Denote $(G, G') \in GI^L_{\mathcal{A}}$, if graph isomorphism algorithm $\mathcal{A}$ decides (G, G') is isomorphic at L^{th} iteration. $GI^L_{\mathcal{A}} \subset GI^L_{\mathcal{B}}$ denotes $\mathcal{A}$ more powerful than $\mathcal{B}$.

Algebra Operation of Matrix Language

conjugate transposition (op$(e) = e^*$)		
$e(\nu(X)) = A \in \mathbb{C}^{m\times n}$	$e(\nu(X))^* = A^* \in \mathbb{C}^{n\times m}$	$(A^*)_{ij} = \overline{A}_{ji}$
one-vector (op$(e) = \mathbf{1}(e)$)		
$e(\nu(X)) = A \in \mathbb{C}^{m\times n}$	$\mathbf{1}(e(\nu(X)) = \mathbf{1} \in \mathbb{C}^{m\times 1}$	$\mathbf{1}_i = 1$
diagonalization of a vector (op$(e) = \text{diag}(e)$)		
$e(\nu(X)) = A \in \mathbb{C}^{m\times 1}$	$\text{diag}(e(\nu(X)) = \text{diag}(A) \in \mathbb{C}^{m\times m}$	$\text{diag}(A)_{ii} = A_i$, $\text{diag}(A)_{ij} = 0,\ i \neq j$
matrix multiplication (op$(e_1, e_2) = e_1 \cdot e_2$)		
$e_1(\nu(X)) = A \in \mathbb{C}^{m\times n}$ $e_2(\nu(X)) = B \in \mathbb{C}^{n\times o}$	$e_1(\nu(X)) \cdot e_2(\nu(X)) = C \in \mathbb{C}^{m\times o}$	$C_{ij} = \sum_{k=1}^{n} A_{ik} \times B$
scalar multiplication (op$(e) = c \times e,\ c \in \mathbb{C}$)		
$e(\nu(X)) = A \in \mathbb{C}^{m\times n}$	$c \times e(\nu(X)) = B \in \mathbb{C}^{m\times n}$	$B_{ij} = c \times A_{ij}$
trace (op$(e) = \text{tr}(e)$)		
$e(\nu(X)) = A \in \mathbb{C}^{m\times m}$	$\text{tr}(e(\nu(X)) = c \in \mathbb{C}$	$c = \sum_{i=1}^{m} A_{ii}$
pointwise matrix multiplication (Schur-Hadamard) (op$(e_1, e_2) = e_1 \odot e_2$)		
$e_1(\nu(X)) = A \in \mathbb{C}^{m\times n}$ $e_2(\nu(X)) = B \in \mathbb{C}^{m\times n}$	$e_1(\nu(X)) \odot e_2(\nu(X)) = C \in \mathbb{C}^{m\times n}$	$C_{ij} = A_{ij} \times B_{ij}$

a_k-walk Message-Passing Framework

Theorem

Given a positive integer sequence a_k and a pair of graphs (G, G'), $S_k = \sum_{t\in[k]} a_t$, if a_k is viewable, then $\forall l \in \mathbb{N}^+$ $GI^l_{a_k-walk} \subseteq GI^{S_l}_{WL}$.

- Given a positive integers sequence a_k, $S_k = \sum_{t\in[k]} a_t$. a_k is *viewable* if $\forall k \in \mathbb{N}^+, r \in [S_k]$(The subset sums of a_k are dense in $\mathbb{N}^+$)
- a_k-walk Message-Passing GNN:

$$M_v^k = \underbrace{AGG(\cdots AGG}_{a_k\ times}(\{\!\{H_u | u \in N(v)\}\!\})). \tag{6}$$

$$H_v^l = COB^l(H_v^{l-1}, M_v^l) \tag{7}$$

Pruned Multi-Aggregation GNN

Theorem

Given a pair of graphs (G, G') and $K \in \mathbb{N}^+$, $\forall L \in \mathbb{N}^+$, for K-Path GNN, the expressiveness of pruned K-Path framework is as powerful as K-Path framework: $GI^L_{PR\,K-P} = GI^L_{K-P}$. For K-Hop GNN, Pruned framework have same expressiveness referrring to regular graphs and strong regular graphs: $(RG \cap GI^L_{PR\,2-H}) \subseteq (RG \cap GI^L_{2-H})$, $(SRG \cap GI^L_{RE\,K-H}) \subseteq (SRG \cap GI^L_{K-H})$.

- Pruned Multi-Aggregation GNN: When $l \leq K$:

$$M_v^{l,k} = AGG_k^l(\{\{H_u^{l-1} | u \in N_{path}^k(v)\}\})(k \leq l \leq K) \tag{8}$$

$$\mathbf{M}_v^l = (M_v^{l,l}, M_v^{l,l+1}, \cdots, M_v^{l,K}) \tag{9}$$

$$H_v^l = COB^l(H_v^{l-1}, \mathbf{M}_v^l) \tag{10}$$

else when $l > K$:

$$\begin{aligned} M_v^{l,K} &= AGG_K^l(\{\{H_u^{l-1} | u \in N_{path}^K(v)\}\}) \\ H_v^l &= COB^l(H_v^{l-1}, M_v^{l,K}) \end{aligned} \tag{11}$$

The Pruning Weisfeiler-Lehman Algorithm

Input: Graph $G = (V, E)$, number of iterations L.

Initialization: $\forall v \in V$, $\chi^0(v)$, $l = 0$.

while $l \leq L$ **do**

  $l = l + 1,\ t = 1.$

  $m_1^l(v) = Hash(\{\{\chi^{l-1}(u) : u \in N(v)\}\}).$

  for $\forall v \in V$ **do**

    while $t < 2^{l-1}$ **do**

      $t = t + 1.$

      $m_t^l(v) = Hash(\{\{m_{t-1}^l(u) : u \in N(v)\}\}).$

    end while

    $\chi^l(v) = Hash(\chi^{l-1}(v), m_{2^{l-1}}^l(v)).$

  end for

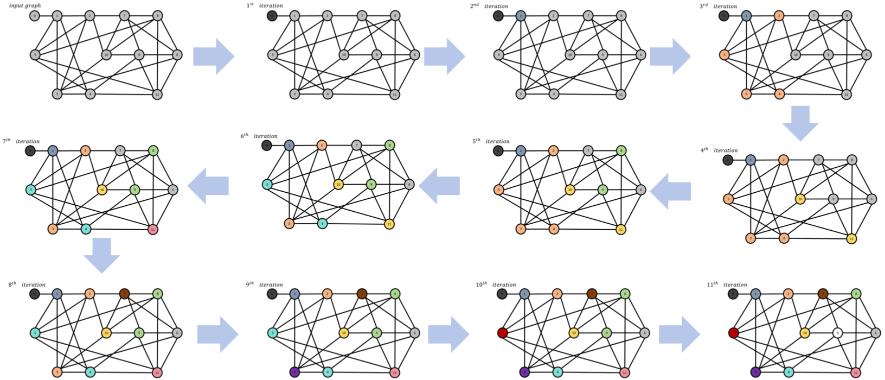
end while

Output: Final labels $\chi^L(v)$ for all $v \in V$.

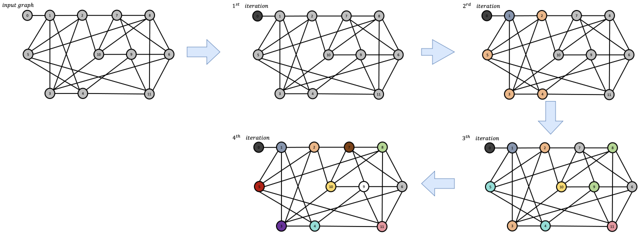
The Pruned Multi-Weisfeiler-Lehman Algorithm

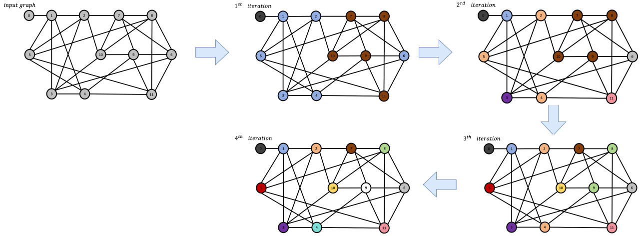
Input: Graph $G = (V, E, X)$, number of iterations L.
Initialization: $\forall v \in V$, $\chi^0(v)$, $l = 0$.
while $l \leq K$ **do**
  $l = l + 1$.
  for $v \in V$ **do**
    for $t \in [l, K]$ **do**
      $\chi_t^l(v) = Hash(\{\{\chi^{l-1}(u) : u \in N_t^t(v)\}\})$.
    end for
    $\mathbf{X}^l(v) = (\chi_l^l(v), \chi_{l+1}^l(v), \cdots, \chi_K^l(v))$.
    $\chi^l(v) = Hash(\chi^{l-1}(v), \mathbf{X}^l(v))$.
  end for
end while
while $l \leq L$ **do**
  $l = l + 1$.
  for $v \in V$ **do**
    $\chi^l(v) = Hash(\chi^{l-1}(v), \{\{\chi^{l-1}(u) : u \in N_{path}^L(v)\}\})$.
  end for
end while
Output: Final labels $\chi^L(v)$ for all $v \in V$.

Process of WL Test on Long-refinement Graph



Process of Pruned WL Test and Pruned 2-hop WL Test on Long-refinement Graph





Comparison on training efficiency

Table: Comparison on training efficiency

Model	COLLAB		NCI1		IMDB-B		IMDB-M		MUTAG		PROTEINS	
	Time	Acc (%)	Time	Acc (%)	Time	Acc (%)	Time	Acc (%)	Time	Acc (%)	Time	Acc (%)
GIN(3)	1.104	74.8 ± 1.3	0.480	71.9 ± 0.5	0.251	71.9 ± 0.3	0.304	49.9 ± 0.0	0.889	89.4 ± 0.4	0.268	73.7 ± 0.7
PR GIN(1)	1.060	73.9 ± 0.0	0.461	72.9 ± 1.4	0.209	69.9 ± 2.0	0.284	50.6 ± 0.3	0.886	88.5 ± 0.0	0.233	72.2 ± 1.9
GIN(7)	1.638	77.4 ± 1.6	0.748	71.5 ± 1.4	0.578	72.6 ± 0.3	0.534	51.1 ± 0.3	0.904	89.4 ± 1.0	0.527	76.3 ± 0.2
PR GIN(124)	1.284	76.4 ± 0.7	0.6961	75.4 ± 0.2	0.481	71.7 ± 1.4	0.464	52.0 ± 0.5	0.916	92.0 ± 0.4	0.425	74.1 ± 1.0
GIN(10)	2.142	74.7 ± 0.6	1.122	75.9 ± 1.3	0.948	72.1 ± 2.8	0.898	49.7 ± 1.5	0.874	87.7 ± 0.2	0.867	72.3 ± 0.0
PR GIN(1234)	1.689	75.6 ± 0.3	0.981	74.7 ± 0.2	0.710	71.5 ± 0.5	0.780	51.2 ± 0.9	0.929	90.7 ± 2.1	0.667	72.5 ± 2.2
2-Hop(3)	1.357	76.8 ± 0.8	0.608	73.6 ± 0.9	0.410	71.0 ± 0.7	0.415	50.1 ± 1.5	0.910	91.0 ± 0.0	0.394	69.5 ± 1.3
PR 2-Hop(3)	1.180	75.1 ± 1.1	0.528	76.5 ± 1.6	0.357	71.5 ± 0.5	0.361	52.5 ± 0.7	0.929	91.3 ± 1.5	0.342	73.3 ± 0.3
2-Hop(5)	1.927	74.2 ± 0.5	0.880	70.6 ± 1.7	0.680	68.8 ± 0.8	0.628	49.5 ± 0.6	0.894	88.3 ± 1.0	0.621	71.0 ± 0.8
PR 2-Hop(5)	1.606	74.5 ± 0.4	0.734	71.1 ± 1.5	0.566	69.7 ± 0.2	0.523	48.0 ± 2.2	0.871	88.7 ± 1.5	0.517	72.0 ± 0.3
2-Path(3)	1.385	75.6 ± 0.0	0.620	73.0 ± 0.8	0.418	71.4 ± 0.1	0.423	49.5 ± 1.8	0.9045	88.7 ± 1.7	0.402	72.7 ± 2.0
PR 2-Path(3)	1.385	76.1 ± 0.3	0.632	75.5 ± 0.4	0.488	74.2 ± 1.9	0.451	51.6 ± 0.4	0.915	91.6 ± 0.0	0.446	76.9 ± 0.7
2-Path(5)	2.111	76.0 ± 0.2	0.964	74.2 ± 0.3	0.745	70.0 ± 0.6	0.688	51.3 ± 0.1	0.890	89.4 ± 0.4	0.680	69.3 ± 1.0
PR 2-Path(5)	1.730	72.1 ± 2.0	0.790	70.5 ± 2.3	0.610	71.8 ± 1.5	0.564	50.2 ± 0.2	0.898	88.9 ± 0.8	0.557	71.9 ± 2.4

Thank You for Listening!